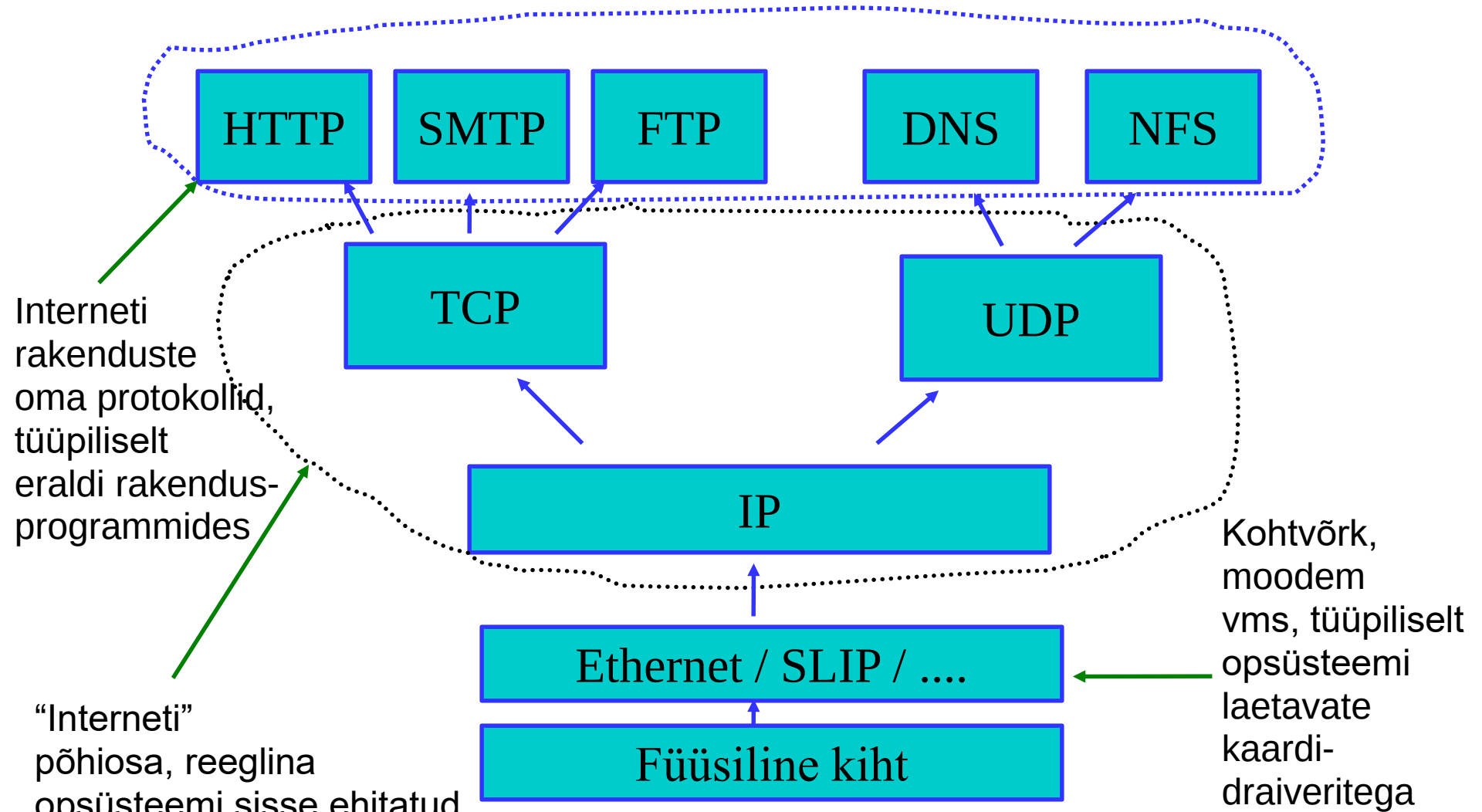
Sissejuhatus infotehnoloogiasse

interneti rakenduste tehnoloogia

Loengu ülevaade. Interneti rakendused.

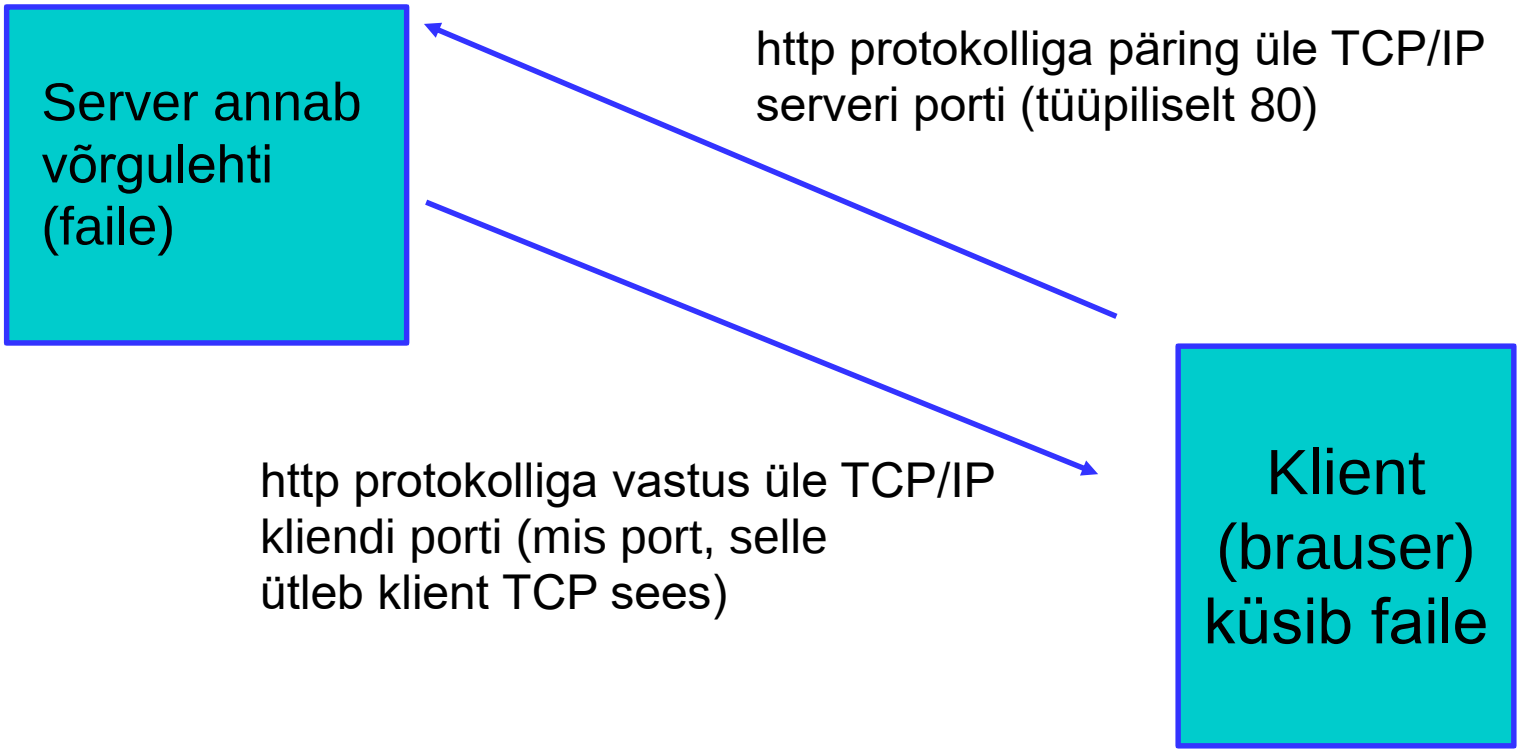
- Meeldetuletus: baasprotokollid, HTTP protokoll
- HTML keel
- Failide tõmbamine ja serveerimine:
  - Kuidas http abil faili tõmmata
  - Info saatmine serverile: cgi protokoll
- Mis brauseril sees on:
  - Html, css, javascript: baastehnoloogiad
  - WebGL, canvas, pisiandmebaasid, xml, xslt ja veel hulk eritehnoloogiaid
- Masintöödeldav andmete esitamine: JSON ja XML
- Andmete tähendus ehk semantika

Kokkuvõte protokollindusest internetis



HTTP ühendused: failide küsimine ja nende andmine

- **HTTP on omaette protokoll**, mida kasutatakse veebilehtede, piltide, tekstifailide, zip failide jne jne saatmiseks veebiserveri ja brauseri vahel.



Html: veebilehtede põhikeel

- Põhimõtteliselt kujunduskeel, mitte programmeerimiskeel
- Hea tutorial: https://www.w3schools.com/html/default.asp
- Veebirakenduste kursus meil:

  http://lambda.ee/wiki/V%C3%B5rgurakendused_I

Tutvume põhimõtetega loengus tehtavate näidete abil

Kuidas html-faili tõmmata?

- Vaatame lihtsat java ja pythoni koodi näidet:

http://java.sun.com/docs/books/tutorial/networking/urls/readingURL.html
https://docs.python.org/2/howto/urllib2.html

- Toorlugemise tööriistad ja brauseritaolised rakendused:
  - wget
  - curl
  - lynx
  - ....

Veebilehtede (failide) serveerimine

- Veebilehed on lihtsalt failid, mille ette server paneb http-päise
- Veebileht (fail) võetakse reeglina kas:
  - olemasoleva failina arvuti kettal
  - tekstina, mille teeb iga kord uuesti mõni programm
    - Programm võib olla külge-ehitatud tava-veebiserverile (php, mod-perl jne)
    - Või käia kohe eri-veebiserveri sees (java tomcat)
    - ... või olla eraldi programm, mille server käivitab (klassikalised cgi-programmid)
    - Kombinatsioonid serverite ahelatega on ka kasutusel

Andmete saatmine serverile: cgi

- URL-i lõpus (peale ?) oleva teksti saab serveri kutsutav programm (nn cgi-programm) kätte
- Brauser saadab vormi info cgi-kodeeritult nii: a=1&b=35 jne, kus a ja b jne on välja nimed, võrduse järel aga kasutaja sisestatud väärtused
- Tüüpiline url cgi parameetritega

http://dijkstra.cs.ttu.ee/~tammet/cgi-bin/loeng/tst6.py?a=1&b=2

http://dijkstra.cs.ttu.ee/~tammet/cgi-bin/loeng/tst8.py?a=12&b=44

- Vt lisaks ja detaile: http://lambda.ee/wiki/Cgi_examples

Brauser oskab mitut keelt

- Iga normaalne brauser mõistab:

Põhiasjad:

- Html
- Css
- Javascript

Lisatehnoloogiad:

- WebGL graafika
- Canvas
- Video ja audio
- Xslt
- Xml
- Väikesed andmebaasid
- ...

Kust lugeda brauseri-tehnoloogiatest

- Ametlikud standardid: http://www.w3.org
- Tutorialid: http://www.w3schools.com
- Tutorialid: http://www.codecademy.com/tracks/web

HTML

- HTML:
  - **Teksti paigutamise / lehe kujundamise keel**
  - Näited kohapeal
  - Loe: http://www.w3schools.com/html/default.asp
  - Eelmine ametlik standard: http://www.w3.org/TR/html401/
  - Uus ametlik standard on **HTML5**:
    https://www.w3.org/TR/html5/
    komplekt uusi kasulikke tehnoloogiaid

- Mis on XHTML:
  - Praktiliselt seesama, mis HTML (samad tagid jne)
  - Tagid peavad olema väikeste tähtedega
  - Peab olema korrektses XML süntaksis (tagi suletud jne)
  - Paar XML-i lisaknihvi ka (namespaced jne)
  - Ei ole eriti mainstream või eriti laialt kasutatud

CSS

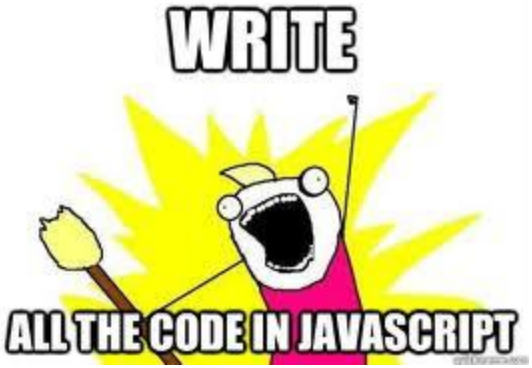
- **Eriti täpset teksti paigutust ja kujundust võimaldav keel HTML täienduseks**

- Näited kohapeal
- Uuri lisaks: http://www.w3schools.com/css/default.asp

Javascript

- **Brauseri programmeerimiskeel: javascripti programmid töötavad otse brauseris: muudavad htmli, css-i, võtavad ühendust serveriga jne jne**

- Näited kohapeal
- Uuri lisaks: http://www.w3schools.com/js/default.asp



Javascript, CSS, asynchronous queries

AJAX tähistab: HTML+CSS+Javascript+asynchronous queries

Mõte selles, et saad brauseris javascriptiga avada serveri urli a la nii

```
fetch("http://dijkstra.cs.ttu.ee/~tammet/cgi-bin/fetch.py", {
   method: "post",
   body: game
 }).then(r=>r.text()).then(handlestore);
```

ja siis brauseris saadud teksti või andmetega (tüüpiliselt javascripti formaadis ehk jsonis) edasi tegutseda.

Tähtis: seni, kui datat tõmmatakse, sinu javascripti programm ei hangu, vaid käib normaalselt edasi!

XML tähendab ...

- **eXtensible Markup Language**

<tootja><nimi>A Le Coq</nimi><linn>Tartu</linn></tootja>

- XML on:
  - Struktureeritud teksti esitamise formaat
  - XML standard ütleb, kuidas teksti struktuuri märgistada.
  - Saab kasutada andmete esitamiseks tekstina
  - Lihtne, aga veidi kohmakas kasutada

- XML ei ole:
  - Programmeerimiskeel.

JSON tähendab ...

- **Javascript Object Notation**

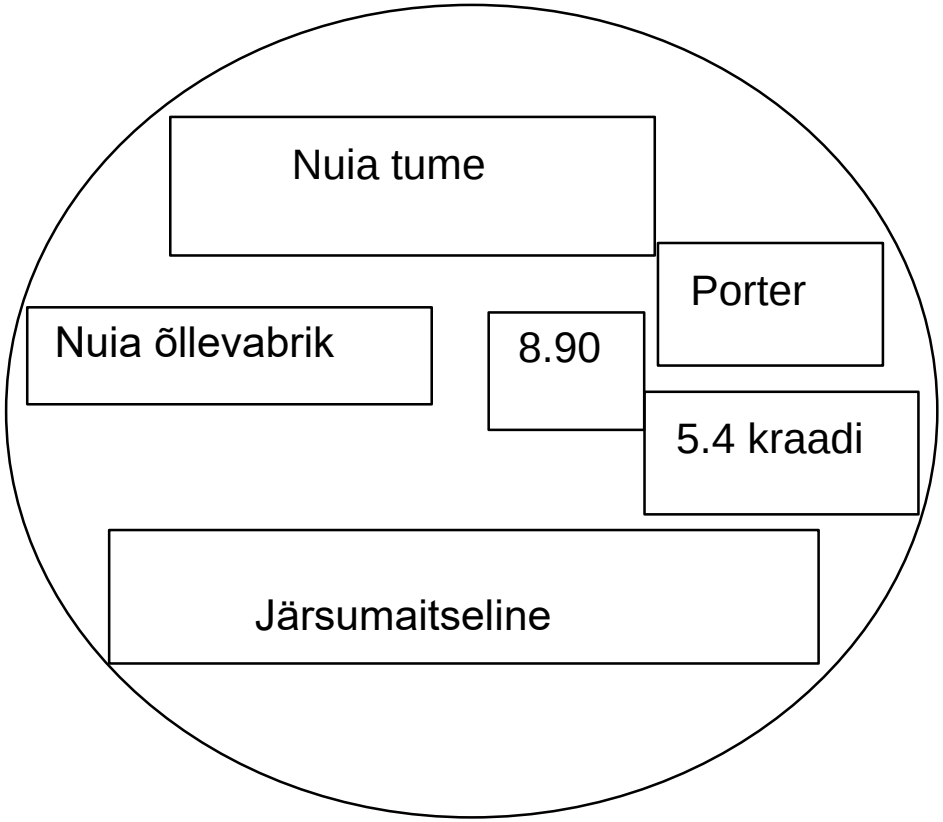
[1,2,“siin on tekst“, [“sisemine“,5], 10]
{“nimi“:“Peeter“, “pikkus“: 180, “hinded“: [5,3,2] }

- JSON on:
  - Andmete esitamise formaat tekstina
  - Javascripti programmeerimiskeele „native“ andmestruktuur
  - Väga lihtne ja mugav kasutada
  - Kaasajal brauserirakendustes eelistatum kui XML.

- JSON ei ole:
  - Programmeerimiskeel.

Tüüpiline probleem andmete esitamisel

- **Meil on mingid andmed.**
- **Neid on vaja faili kirja panna ja hiljem arvutiga töödelda ja teistele edasi saata.**
- Olgu meil info mingi õllesordi kohta:



Kuidas seda faili kirja panna?

- **Variant 1: inimkeelsena.**

  Nuia tumeda tootja on Nuia õllevabrik. Porteri tüüpi õlu. Kangus 5.4 kraadi. Järsumaitseline. Hind 8.90.

  - Oleks hea variant, aga arvutiga praktiliselt lootusetu töödelda.

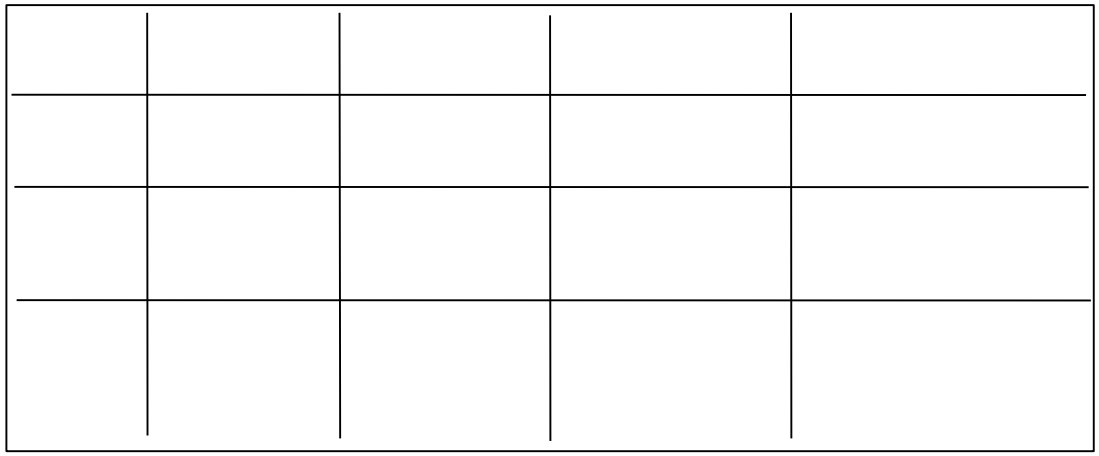
- **Variant 2: CSV (comma separated values)**

  “Nuia tume”, “Nuia õllevabrik”, Porter, 8.90, 5.4, Järsumaitseline

  - Päris hea ja väga levinud variant.
  - Failis ei saa öelda, mis tähestikus (õ, ä jne kodeering).
  - Vaja on eraldi öelda, mida väljad tähendavad (selleks saab kasutada päiserida).
  - Põhiprobleem: järgmisel slaidil

CSV ja muude tabelkujude põhiprobleem

- **CSV ja muud tabeliformaadid esitavad infot tabelina:**



- See ei sobi hästi, kui tabeli ruutudes peaks olema alamtabelid.
- Näiteks, kui:
  - Tahaksime sisestada aadressi: riik, linn/küla, tänav, maja/krt
  - Hinna juures on hinnaühik (EEK, EUR, jne)
  - Jne
- SQL andmebaasides kasutatakse siis abitabeleid ja neid siduvaid kunstlikke ID-numbreid, mis pole seotud algse dataga.

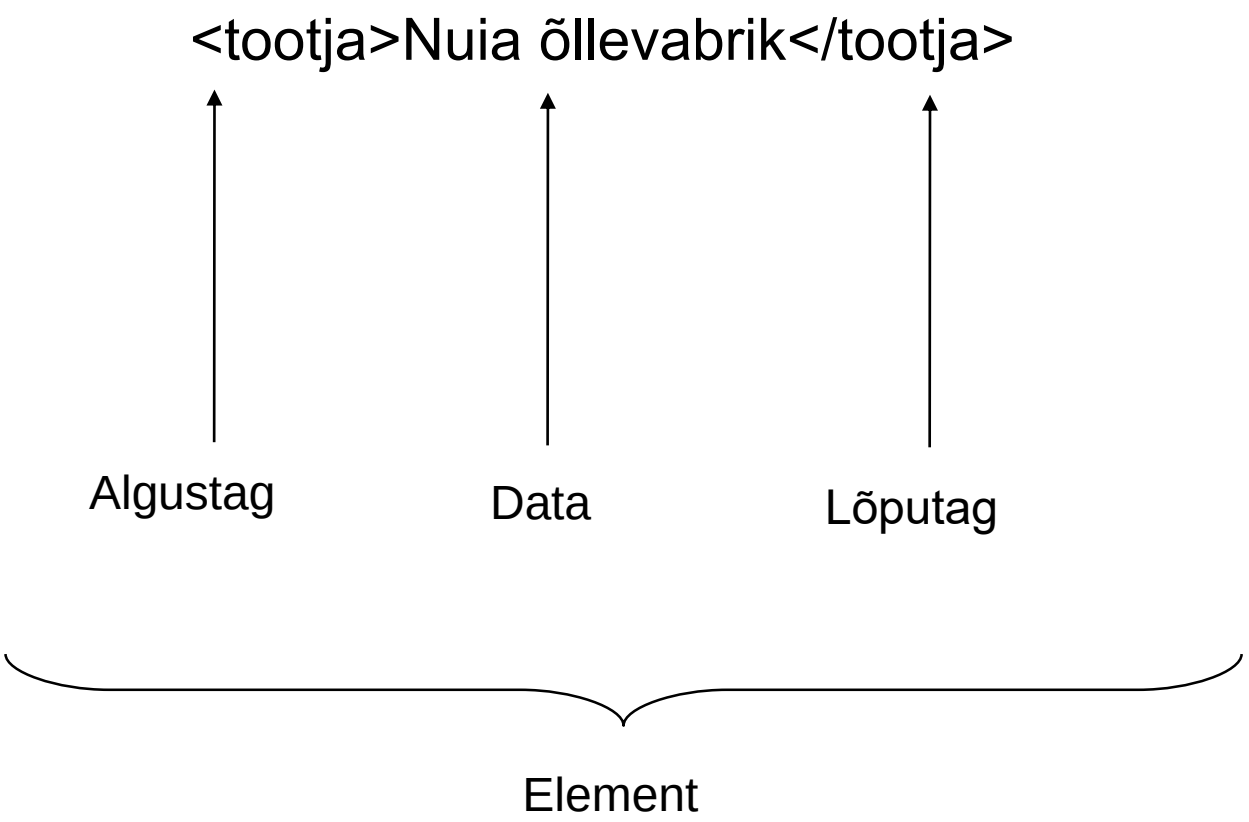
XML viis eelmist infot kirja panna

- **XML-s “pakitakse” infoväljad kirjeldavate nn tag**-de vahele:

```
<olu>
  <nimi>Nuia tume</nimi>
  <tootja>Nuia õllevabrik</tootja>
  <tyyp>Porter</tyyp>
  <hind>8.90</tyyp>
  <kangus>5.4</kangus>
  <maitse>Järsumaitseline</maitse>
<olu/>
```

- Pluss: saab töödelda arvutiga nagu CSV-d
- Pluss: inimesele kohmakas lugeda, kuid siiski üheselt arusaadav.
- Miinus: tekst on pikem, kui CSV puhul.
  - Sellest saab üle: kokkusurumine (zip, gz, jms pakkijad).

XML tagid


<tootja>Nuia õllevabrik</tootja>
Algustag
Data
Lõputag
Element

Tagide põhieelis

- Elemendid võivad sisaldada teisi elemente:

```
<olu>
    <nimi>Nuia tume</nimi>
    <tootja>
        <nimi>Nuia õllevabrik</nimi>
        <aadress>
            <küla>Nuia</küla>
            <tänav>Tallinna mnt</tänav>
            <maja>2</maja>
        </aadress>
    </tootja>
    <tyyp>Porter</tyyp>
    <hind>8.90</tyyp>
    <kangus>5.4</kangus>
   <maitse>Järsumaitseline</maitse>
<olu/>
```

HTML ja XML

- Info pannakse “tag”-ide vahele: infol on sildid

- HTML: “tag”-del visuaalne semantika (st tähendus)

  ****Siin on paks tekst****
  ****Siin on kaldkirjas tekst****

- XML: “tag”-del semantika puudub

  <autor>Tanel Tammet**</autor>**
  <minuaadress>Kuiv 9**</minuaadress>**

JSON ja XML

- Eelmisel kümnendil muutus XML väga populaarseks andme-esituskeeleks.
- Seejärel hakati kasutama hulka keerulisi lisatehnoloogiad: XSLT, SOAP, XML Schema, UDDI jne, mis muutsid asja keeruliseks.
- Seejärel muutus populaarseks JSON kui hulga lihtsam andme-esituskeel: lihtsalt javascripti andmesüntaks, väga sarnane pythoni süntaksile a la

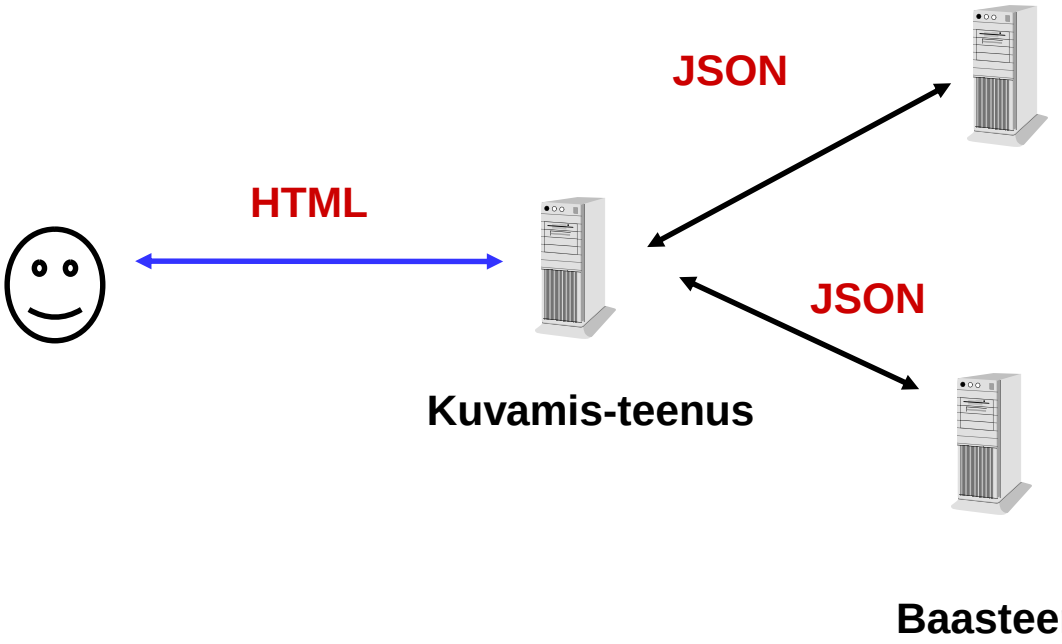
  {“nimi“: “Tanel“, “kursused“: [“itv0010“, “itv100“]}

- Praegu kasutatakse mõlemat, aga JSON on palju populaarsem. Laiatarbe-süsteemid a la Google pakuvad enamasti JSON-is andme-esitust

Võrguteenused

- Probleem: raske on teha programme, mis loeks teisest serverist HTML-lehekülgi ja leiaks nende sisust kergesti vajaliku info

- Lahendus: teeme võrgulehekülgi lihtsalt andmetekstina jsonis või xmlis nii et programm teises arvutis suudaks neid lugeda



Semantiline veeb: w3c vana suur projekt

- Seni ei ole läinud nii hästi, kui loodeti.
- Eesmärk: pakkuda andmete esitamise keeli ja luua standardeid, mille abil kirjeldada asju (internetis)
- Soovitav tulemus: saab teha tarkvara, mille abil datat saab kergesti ühest programmist teise saata, nii et info ja mõte kaduma ei lähe.
- Esmajoones vaja:
  - objektide kirjeldamise keel
  - mõistete omavaheliste seoste ja reeglite keel
- Projekti juht: Tim Berners-Lee (http, www ja esimese brauseri autor, w3c juht).

Semantic webi planeeritud kihiline süsteem

Tõestuste keel

Täisloogika

Ontoloogiad

Ontoloogiakeel (Owl ja tema alamhulgad)

Asjade kirjelduskeel (RDF ja RDFS)

Nimede unikaliseerimine (namespaces)

Süntaks (XML, JSON, turtle, ...)

Transport (http, https, SOAP, tcp/ip ...)

Mis on keeruline osa semantilise veebi projektis?

- Üks programm peab teise programmi antud XML-ist SISULISELT aru saama
- Õllepartiide otsimise server:

Hulgimüüja A annab infot:

```
<beer>
  <name>Guinness</name>
  <price>100</price>
</beer>
```

Hulgimüüja b annab infot:

```
<olu>
   <mark>Guinness</mark>
   <hind>100</hind>
</olu>
```

Hulgimüüja C annab infot:

```
<porter>
  <name>Guinness</name>
  <price>100</price>
</porter>
```

Erinevad keeled

- Peame teadma, et:

BEER	=	ÕLU
PORTER	on ka	BEER

- Ei ole lootust teha “ette valmis” kõiki keeli
- On olemas alamkeeled (üldine toodete keel, õllede keel Eestis, Tshehhis, Hiinas, õllede keel Tartu Õllevabrikus ja Sakus, ...)
- Keeli on vaja omavahel tõlkida
- Tõlkimiseks on vaja reegleid ja elementaarset arusaamist

Asjade kirjeldamine: RDF

- Vaata

https://courses.cs.ttu.ee/pages/Teadmisp%C3%B5hise_tarkvaraarenduse_meetodid_/_Methods_of_Knowledge_Based_Software_Development_2017#Knowledge_representation

- Idee: kõik kirjeldused on kolmikud: [objekt, omadus, väärtus] a la

client_1	name	„John Brown“
client_1	balance	200
client_1	customer	client_2
client_2	name	„Mike“

Võrguteenused ja RDFa

- Näide htmli semantilisest annoteerimisest

Algse html-teksti

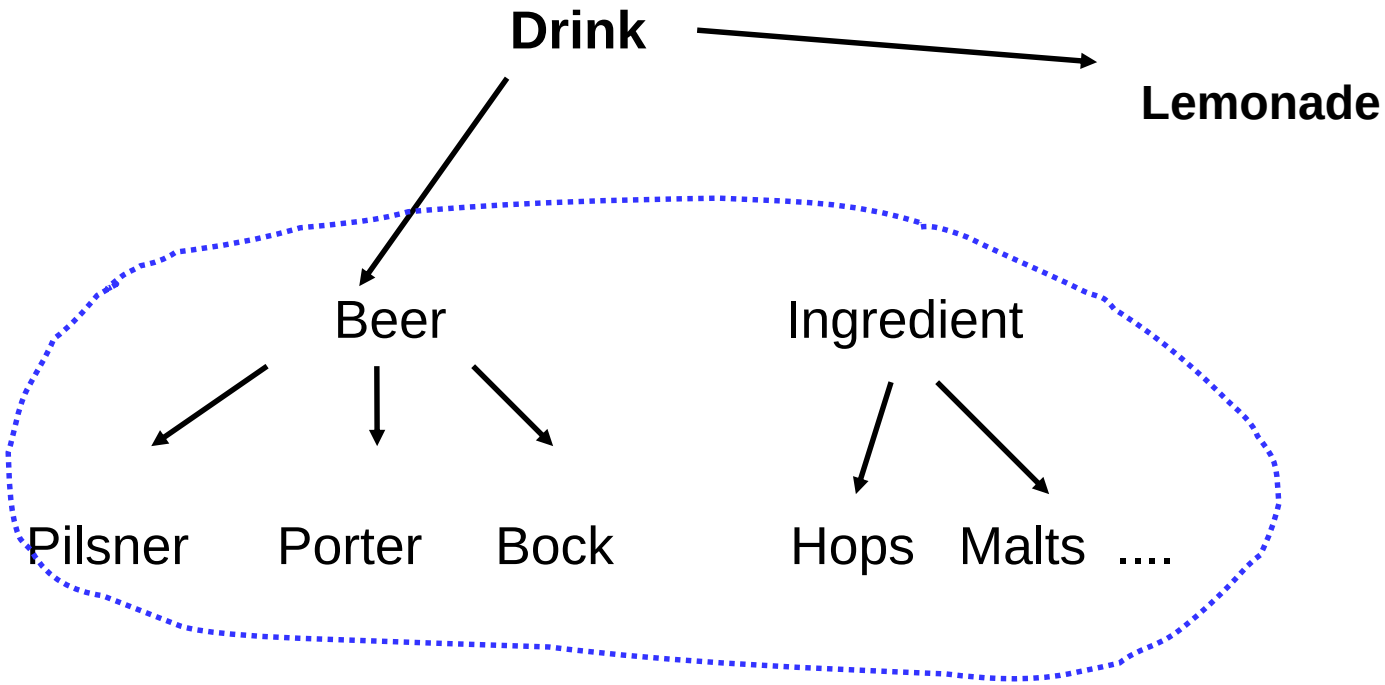
```
<tr>
   <td>Jaanus Kask</td>
   <td>6024554</td>
   <td><a href="http://kask.googlepages.com">kliki siia</a></td>
</tr>
```

asemel on siin masinloetavaks html-tekstiks RDFa järgi annoteeritult

```
<tr about="#jaanus_kask">
   <td property="er:nimi">Jaanus Kask</td>
   <td property="er:telefon">6024554</td>
   <td><a rel="er:koduleht"
          href="http://kask.googlepages.com">kliki siia</a></td>
</tr>
```

Ontoloogiad ehk mõistete hierarhiad ja lihtseosed

- Vaata http://schema.org
- Vaata https://developers.google.com/search/docs/guides/intro-structured-data



- Eri valdkondades on oma ontoloogiad

- Ei peagi panema kõiki programme kasutama samu “standardontoloogiaid”